Discrete Applied Mathematics 156 (2008) 3506–3509

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Note

Exponential behaviour of the Butkovič–Zimmermann algorithm for solving two-sided linear systems in max-algebra

Marc Bezem [a], Robert Nieuwenhuis [b,*], Enric Rodríguez-Carbonell [b]

[a] Department of Informatics, University of Bergen, Bergen, Norway

[b] Technical University of Catalonia, Barcelona, Spain

ARTICLE INFO

Article history:
Received 23 October 2007
Received in revised form 22 February 2008
Accepted 8 March 2008
Available online 22 April 2008

Keywords:
Max-algebras
Two-sided equations
Polynomial algorithm

ABSTRACT

In [P. Butkovič, K. Zimmermann, A strongly polynomial algorithm for solving two-sided linear systems in max-algebra, Discrete Applied Mathematics 154 (3) (2006) 437–446] an ingenious algorithm for solving systems of two-sided linear equations in max-algebra was given and claimed to be strongly polynomial. However, in this note we give a sequence of examples showing exponential behaviour of the algorithm. We conclude that the problem of finding a strongly polynomial algorithm is still open.

1. The problem under consideration

Max-algebras naturally arise in many contexts, such as decision theory, discrete event dynamic systems, and operations research.[1] Here we consider the same problem as in [1], namely solving systems of two-sided linear equations in max-algebra. More precisely, we consider systems of equations over a given set X of n variables, denoted here by $\{x_1, \dots, x_n\}$, where each equation has the form:

$$\max(x_1 + a_1, \dots, x_n + a_n) = \max(x_1 + b_1, \dots, x_n + b_n),$$

with $a_1, \dots, a_n, b_1, \dots, b_n \in \mathbb{Q}$. The a_i and b_i are called *offsets* and the $x_i + a_i$ and $x_i + b_i$ are called *terms*.

The aim here is to either find a solution (i.e. rational values for the variables of X, such that all equations hold under the usual interpretations of max and $+$) or to decide that no such a solution exists.

In [1], a very elegant and ingenious algorithm for doing this is given and claimed to be strongly polynomial. This would solve a problem with important practical applications which has been open for more than 30 years. Unfortunately, in this note we give a sequence of counterexamples showing exponential behaviour of that algorithm. We conclude that the problem of finding a polynomial algorithm is still open.

Ref. [1] initially considers rational variables and offsets, but their algorithm can also handle other algebraic structures, including the integers. The construction of the counterexample we give in this note applies to the other structures as well.

* Corresponding author. Fax: +34 93 413 7833.
E-mail address: roberto@lsi.upc.edu (R. Nieuwenhuis).
URLs: http://www.ii.uib.no/~bezem (M. Bezem), http://www.lsi.upc.es/~roberto/~erodri (R. Nieuwenhuis).

1 See, e.g., ralyx.inria.fr/2006/Fiches/maxplus/maxplus.html.

0166-218X/$ – see front matter
doi:10.1016/j.dam.2008.03.016

2. The algorithm of [1]

Here we only give a short intuitive description of the algorithm; for details, see [1]. Let E denote the given system of equations and let the (possibly subscripted or primed) symbol S denote *states* of the algorithm, i.e., functions $S : X \to \mathbb{Q}$.

It is easy to see that if a state S is a solution for E, then, for any rational constant c, so is the state S' defined as $S'(x) = S(x) - c$ for all x. Therefore, the algorithm can start in an arbitrary initial state and from then on only search solutions among states obtained by decreasing values of variables. This is done in such a way that currently false equations may become true, while true equations remain true.

Fixpoint construction of MD(S, E), the set of variables that Must Decrease.

Let the current state S be the *all-zero* state S_0 where $S_0(x) = 0$ for all $x \in X$, and consider a set E with variables x, y, z and u:

$$\max(x, y, z+1, u) = \max(x+5, y+5, z, u) \tag{1}$$

$$\max(x, y-3, z-4, u-1) = \max(x-2, y-2, z, u-5) \tag{2}$$

$$\max(x, y-2, z-2, u-2) = \max(x, y-2, z-2, u-1). \tag{3}$$

Every currently false equation (here, only (1)) forces to decrease one or more variables. In this case, x and y must decrease, i.e., $x, y \in MD(S, E)$, due to the maximal terms $x + 5$ and $y + 5$ at the right-hand side of (1). Decreasing a variable may force other variables to decrease as well in order to avoid that true equations become false. For example, (2) is true in S, but if $x \in MD(S, E)$, then $z \in MD(S, E)$ is forced to keep (2) true. Due to other true equations (not shown here), $z \in MD(S, E)$ may force $u \in MD(S, E)$, etc. This is iterated until no more variables are added to $MD(S, E)$, giving a polynomial-time fixpoint construction of $MD(S, E)$, since at most $|X|$ variables are added.

Determining the decrement τ.

Once $MD(S, E)$ has been identified, all variables in $MD(S, E)$ are decreased by the same amount τ, that is, we obtain a new current state S' with $S'(x) = S(x) - \tau$ if $x \in MD(S, E)$ and $S'(y) = S(y)$ otherwise. The value of τ is essentially the minimal amount such that $MD(S, E) \neq MD(S', E)$. This can be due to three reasons:

(i) Because some false equation becomes true in S'; for example, in S_0 the equation (1) becomes true in S' if $\tau = 4$.

(ii) Because a certain true equation is no longer a reason for decreasing a variable. For instance, consider (2) in S_0. After decreasing x and z with $\tau = 1$, the variable x can continue decreasing without z, because of the term $u - 1$ at the left-hand side of (2).

(iii) Because in S' some true equation causes an additional variable to be added to the set. For example, in S_0, after decreasing x by 1, the true equation (3) causes u to belong to $MD(S', E)$.

It is not hard to prove that E has no solution if for some S, such a τ does not exist. This includes the case where $MD(S, E) = X$.

The algorithm.

The algorithm of [1] iterates these two steps: computing $MD(S, E)$ for the current S, determining τ, thus obtaining a new S, and so on, until either all equations become true (i.e. the current S is a solution), or τ does not exist, and hence E has no solution.

3. Exponential behaviour of the algorithm

An algorithm is (strongly) polynomial if there exists a polynomial function P such that for every input I its runtime is below $P(size(I))$ (where *size* refers to the number of bits). Below we give a sequence $E_0, E_1, E_2, \ldots$ of input systems where for each E_i its size is polynomial in i (essentially cubic) but where the runtime of the algorithm of [1] is exponential in i, namely at least 2^i. This implies that the algorithm is not polynomial: for every polynomial P there exists a large enough i such that $2^i > P(i^3)$. In the following, we will write states as tuples of values of the form $(v_1, \ldots, v_n)$ for the variables $(x_1, \ldots, x_n)$.

System E_0 consists of the single equation $\max(x_0 - 1, y_0) = \max(x_0 - 1, y_0 - 4)$ over two variables, with the initial state $(x_0, y_0) = (-1, 0)$. Here $MD(S, E_0) = \{y_0\}$, with $\tau = 2$ and the algorithm terminates after one step in state $(-1, -2)$. Since no variable becomes lower than -2 and in all equation sides there is at least one offset -1 or higher, the terms with offset -4 will never become maximal and are hence irrelevant in the algorithm. This kind of irrelevant offsets will be called the *irrelevancy offset* of the system. In what follows, we will omit in equations all terms with the irrelevancy offset. E_0 then becomes $\max(x_0 - 1, y_0) = \max(x_0 - 1)$.

For $i > 0$, the system E_i is always obtained from E_{i-1} by:

1. Taking system E_{i-1}, but doubling all offsets (including the irrelevancy one) and the initial values, and adding two more variables x_i and y_i with the (doubled) irrelevancy offset. The initial values for (x_i, y_i) are always $(-1, 0)$.
2. Making x_i behave in the algorithm as $\min(x_{i-1}, y_{i-1}) + 1$ and making y_i behave as $\max(x_{i-1}, y_{i-1})$, by adding the following three equations:

$$\max(x_{i-1} + 1) = \max(x_{i-1} + 1, x_i)$$
$$\max(y_{i-1} + 1) = \max(y_{i-1} + 1, x_i)$$
$$\max(y_i) = \max(x_{i-1}, y_{i-1}).$$

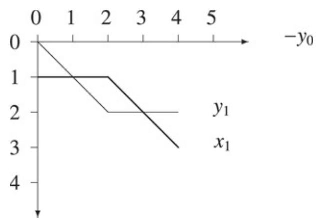
Hence, system E_1 is:

(0) $\max(x_0 - 2, y_0) = \max(x_0 - 2)$
(1a) $\max(x_0 + 1) = \max(x_0 + 1, x_1)$
(1b) $\max(y_0 + 1) = \max(y_0 + 1, x_1)$
(1c) $\max(y_1) = \max(x_0, y_0)$

with irrelevancy offset -8 and with initial values $(x_0, y_0, x_1, y_1) = (-2, 0, -1, 0)$. The algorithm runs in 2 iterations, where τ is always 2. In the table below we summarize its behaviour, writing between parentheses the number of the relevant equation:

iteration	MD		τ	x_0	y_0	x_1	y_1
initial state :				-2	0	-1	0
1	y_0 (0)	y_1 $(1c)$	(1)	-2	-2	-1	-2
2	y_0 (0)	x_1 $(1b)$	(0)	-2	-4	-3	-2

The following figure shows the evolution in the course of the algorithm on E_1 of the variables x_1 and y_1 as functions of $-y_0$:



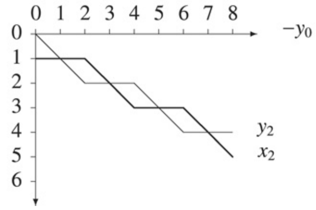
In this figure we see that x_1 and y_1 *cross* (change order) twice. The key idea behind our sequence of counterexamples is that for each E_i the number of such crossings between x_i and y_i is doubled with respect to the number of crossings between x_{i-1} and y_{i-1} in E_{i-1}. The reason for this is precisely that each time x_i is $\min(x_{i-1}, y_{i-1})$ plus a small amount and that y_i is $\max(x_{i-1}, y_{i-1})$.

System E_2 is:

(0) $\max(x_0 - 4, y_0) = \max(x_0 - 4)$
(1a) $\max(x_0 + 2) = \max(x_0 + 2, x_1)$
(1b) $\max(y_0 + 2) = \max(y_0 + 2, x_1)$
(1c) $\max(y_1) = \max(x_0, y_0)$
(2a) $\max(x_1 + 1) = \max(x_1 + 1, x_2)$
(2b) $\max(y_1 + 1) = \max(y_1 + 1, x_2)$
(2c) $\max(y_2) = \max(x_1, y_1)$

with irrelevancy offset -16 and where the initial values for $(x_0, y_0, x_1, y_1, x_2, y_2)$ are $(-4, 0, -2, 0, -1, 0)$. The algorithm runs in 4 iterations, where τ is always 2. The table and graphic below summarize its behaviour, and we see that x_2 and y_2 indeed cross four times:

iteration	MD			τ	x_0	y_0	x_1	y_1	x_2	y_2
initial state :					−4	0	−2	0	−1	0
1	y_0 (0)	y_1 (1c)	y_2 (2c)	(2c)	−4	−2	−2	−2	−1	−2
2	y_0 (0)	y_1 (1c)	x_2 (2b)	(1c)	−4	−4	−2	−4	−3	−2
3	y_0 (0)	x_1 (1b)	y_2 (2c)	(2c)	−4	−6	−4	−4	−3	−4
4	y_0 (0)	x_1 (1b)	x_2 (2a)	(0)	−4	−8	−6	−4	−5	−4



Consider a pair of variables in the algorithm, say x and y. Each time x and y cross (at some point that is not the initial state) it is because one of them is decreasing and the other one is not. The next time they cross, it is the other way around. Hence, between any two of these crossings, at least one new iteration must have started. Since the first iteration starts in the initial state, i.e. before the first crossing of the given x and y we consider, we may conclude that there are at least as many iterations as crossings between x and y. Since each E_i has $3i + 1$ equations, $2i + 2$ variables, and all offsets have size linear in i, we can therefore conclude the following.

Theorem. *For every natural number $i \geq 0$, there exists a two-sided linear system in max-algebra whose size in bits is cubic in i on which the algorithm of* [1] *needs at least 2^i iterations.*

We remark that the observed exponential behaviour is independent of the chosen initial state, which in our example for each E_i is $(x_0, y_0, x_1, y_1, \ \ldots \ x_i, y_i) = (-2^i, 0, \ -2^{i-1}, 0, \ \ldots \ -1, 0)$. Indeed, given another initial state, the system E_i can be replaced by another system E_i' on which the algorithm with the new initial state behaves exactly as E_i did with our initial state. This is easy to verify: if the initial state value k for a variable x is replaced by $k + k'$, then it suffices to decrease in all equations the offset of x by k'.

4. Final remarks

The algorithm of [1] is correct, and we believe it is also *weakly polynomial*, i.e. polynomial not in the size of the input, but in the numerical value of the input, which may be exponentially larger. However, given the simplicity of our example and the intuition acquired by it, we think that finding a polynomial algorithm will require a rather different approach.

Acknowledgements

Partially supported by Spanish Min. of Educ. and Science through the LogicTools project (TIN2004-03382).

References

[1] P. Butkovič, K. Zimmermann, A strongly polynomial algorithm for solving two-sided linear systems in max-algebra, Discrete Applied Mathematics 154 (3) (2006) 437–446.